AFOSR-TR-97 0394

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	20 May 1997	Final Technical April 1994 to March 1997

4. TITLE AND SUBTITLE

Electromagnetic Codes in Complex Geometries

5. FUNDING NUMBERS

AFOSR Grant
F49620-94-0218

6. AUTHOR(S)

Daniel W. Swift

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Geophysical Institute
University of Alaska
Fairbanks, AK 99775-7320

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air force office of Scientific Research/NM
110 Duncan Avenue, Suite B115
Bolling AFB DC 29332-0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION AVAILABILITY STATEMENT

Unlimited

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The project objective is the development of codes to simulate the interaction of charged particles and the electromagnetic field inside of complex chambers. This project is applicable to simulation of microwave devices. The codes are built around the use of generalized curvilinear coordinate systems based on hexahedral elements that can be made to conform to complex boundary surfaces. The new element in this project is the use of multiple coordinate patches to accommodate discontinuous and cylindrical surfaces.

The main accomplishment is the development of techniques to allow integration of the Maxwell equations across coordinate discontinuities. The technique is demonstrated on a solid cylinder of variable cross section. The report also presents grid generation techniques.

14. SUBJECT TERMS

Electromagnetics, Microwaves, Numerical Simulation
Particle-In-Cell Code, Curvilinear Coordinates

15. NUMBER OF PAGES

27 Pages

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	U	U	UL

Final Technical Report
AFOSR Grant F49620-94-0218

Electromagnetic Codes in Complex Geometries

Daniel W. Swift, PI
Geophysical Institute, University of Alaska
Fairbanks, AK 99775-7320
swift@gi.alaska.edu

Project Summary

The project objective is the development of codes to simulate the interaction of charged particles and the electromagnetic field inside of complex chambers. This project is applicable to simulation of microwave devices. The codes are built around the use of generalized curvilinear coordinate systems based on hexahedral elements that can be made to conform to complex boundary surfaces. The new element in this project is the use of multiple coordinate patches to accommodate discontinuous and cylindrical surfaces.

The main accomplishment is the development of techniques to allow integration of the Maxwell equations across coordinate discontinuities. The technique is demonstrated on a solid cylinder of variable cross section. The report also presents grid generation techniques.

(1) Project Objectives

See project summary and text of attached narrative

(2) Status of Research Effort

See project summary and text of attached narrative.

(3) Publications

Swift, D.W., Use of a hybrid code to model the Earth's magnetosphere, *Geophys. Res. Lett.*, **22**, 311, 1995*

Swift, D. W., Use of a hybrid code for global-scale plasma simulation, *J. Comp. Phys.*, **126**, 109 1996*

Lin, Y and D. W. Swift, A two-dimensional hybrid simulation of the magnetotail reconection layer, *J. Geophys. Res.*, **101**, *19,871*, 1996

Lin, Y., D. W. Swift and L. C. Lee, Simulation of pressure pulses in the bow shock and magnetosheath driven by variations in interplanetary magnetic field direction, *J. Geophys. Res.*, **101**, 27,251, 1996

Swift, D. W. Electromagnetic and hybrid codes in multiple-patch coordinate systems, *J. Computer Physics Com.* (In preparation)

* Papers acknowledging AFOSR support

(4) Personnel Associated with Project

Peter Delemere, Ph.D. graduate student. Expected date of graduation: December 1997. Thesis title: *Hybrid code simulation and analysis of CRRES G1, G9 and G11A barium releases.*

(5) Interactions

June 1994: P. C. Liewer and S Karmesin visited Geophysical Institute in Fairbanks, AK to discuss project.

December 1994: Attended American Geophysical Union Meeting in December 1994. Presented paper *Use of a hybrid code to model the Earth's magnetosphere*

December 1994: Attended workshop on *Adaptive Grid Methods for Fusion Plasmas* sponsored by ASIC, NERSC and NYU held in Pleasanton, CA. Presented a paper *Use of Generalized Curvilinear Coordinates Systems in Hybrid and Electromagnetic Codes.*

December 1994: Visited JPL in Pasadena, CA to discuss project with P. C. Liewer and S. Karmesin

July 1995 Attended IUGG meeting in Boulder Colorado and presented a paper Prospects for a *Space-Weather Forecasting Model*

September 1995 Attended Cray User Group conference in Fairbanks, AK and presented a paper *Particle-in-Cell Electromagnetic Codes in Complex Geometries*

December 1995 Attended American Geophysical Union Meeting. Presented paper

December 1995 Visited JPL in Pasadena, CA to discuss project with P. C. Liewer and S. Karmesin

September 1996: Attended Workshop "Encounter between Global Observations and Models in the ISTP Era". Presented a paper entitled *The effect of the Interplanetary Magnetic Field on the Dayside Magnetosphere*

(6) Other

The codes developed have to a large extent solved the problem of simulation of electromagnetic waves inside a cylindrical domain that includes the cylindrical axis. The code at present exists in the form of modules written in Fortran 90. The investigator would be pleased to make the code available to and work with anyone who has an interest in using it. It is often difficult to integrate modules written by one party into code written by somebody else. This investigator also stands willing to collaborate with others who might simply wish to use the algorithms described in this report in another code.

1. Background and Motivation

The original objective of this project had been to apply recently developed algorithms relating to the use of generalized curvilinear coordinates to simulate the interaction of charged particles and the electromagnetic field within microwave generation devices of complex shapes. Curvilinear coordinates enable coordinate surfaces to conform to continuous boundary surfaces, which greatly simplifies the imposition of boundary conditions. Microwave devices involve the interaction between charged particles and the electromagnetic field. This interaction is most easily treated in a coordinate using hexahedral cells, which are topologically equivalent to a cubic lattice. Madsen [1995], on the other hand, described methods simulating electromagnetic propagation using more general finite elements, but his work did not include particle interaction.

Most devices of practical interest have sharp edges requiring discontinuous coordinate surfaces. It became apparent that attempts to approximate a discontinuous surface by a coordinate surface with large curvature would introduce unacceptable errors in the evaluation of the field derivatives needed in Maxwell equations. Eastwood et. al. [1995] show that complex devices can be modeled by use of multiple coordinate patches, each of which may be continuous. The sharp edge of the bounding domain would then lie on the surface separating the coordinate patches. The boundary conditions can be made homogeneous and imposed on a continuous surface. These considerations have led to the focus of this project on the design of a code which accommodates an arbitrary number of coordinate patches, each of which may be dimensioned differently.

Although individual coordinate patches may be continuous, it is not always possible to join the coordinate patches so that grid across the patches is continuous. A good example is the common problem of propagation in a cylinder. An obvious choice would be the use of cylindrical coordinates. The problem is that cylindrical coordinates have a coordinate singularity along the central axis. Figure 1 shows the coverage of the interior of a circular domain by five coordinate patches, similar to that suggested by Eastwood et al., [1995]. Figure 2 shows the use of this coordinate system inside a solid cylinder of arbitrary cross section. Each coordinate patch is continuous, but the coordinates undergo a discontinuous change in direction across the boundaries separating the peripheral patches. The time stepping of Maxwell's equations for the electromagnetic field involves the numerical differentiation of vector fields across coordinate patches. This poses special problems because field components with respect to one coordinate system will be different in another. Another problem is that coordinate systems cannot always be constrained to be orthogonal. The result of a *curl* operation will give a vector component perpendicular to a cell face. Evaluation of a *curl* by Stokes' theorem requires vector components parallel to cell edges. In a non-orthogonal coordinate system cell edges are not parallel to cell face normals. Conversion from one to another requires interpolation, which is a multi-grid point operation that may cross coordinate patch boundaries.

The main focus of efforts reported in this document is the development of robust techniques for accurate numerical integration of Maxwell's equations across discontinuous coordinate patch boundaries. The next section provides a more detailed description of the electromagnetic code and particle mover in curvilinear coordinates. This will provide background for Section 3, which describes the multiple-patch code. The focus will be on

conversion from face-normal to edge tangent components used in the evaluation of *curl***E** and *curl***B** in the neighborhood of and across patch boundaries. The other critical element described in Section 3 will be the particle mover and current algorithm in relation to patch boundaries. Section 4 will describe techniques used for coordinate generation.

2. The Electromagnetic Code in Generalized Curvilinear Coordinates

This section describes the electromagnetic code as implemented for a single coordinate patch. The details of curvilinear methods, as applied to a hybrid code, have been previously reported [Swift, 1996]. The main purpose of this section is to acquaint the reader with the methods that will form the context of the new work to be reported in the next section.

Figure 3 shows the arrangement of the data elements on a hexahedral grid. This layout is analogous to the two-dimensional layout for an electromagnetic code in a rectangular grid described by Birdsall and Langdon [1985]. The important feature to notice in the figure is that the fields are laid out on a staggered grid. The divergence of the electric field lies on the main set of grid points. From Gauss' law, this must also be the locus of the charge, and hence particle, density. This grid point denoted by the integer indices is at the center of the E-cell. The electric field components are located on the faces of the cell, as shown in the figure. The divergence of the magnetic field is located at the half-grid point and is the center of the B-cell. The magnetic field components are similarly located at the faces of the B-cell. The grid point locations of the various curvilinear components of the electromagnetic field are also indicated in figure 3.

2.1 The Field Equations

The function of the electromagnetic code is to advance the two Maxwell equations

$$\frac{\partial \mathbf{B}}{\partial t} = -\nabla \times \mathbf{E}$$
$$\frac{\partial \mathbf{E}}{\partial t} = c^2 \nabla \times \mathbf{B} - \frac{4\pi e^2}{m}\mathbf{J} \qquad (1)$$

Where **B** is defined in terms of the gyrofrequency of a particle of mass m, **E** is defined in terms of the acceleration and **J** is in units of particle flux. Stokes theorem is used to discretize these equations. For example, Faraday's law takes the form

$$\int \frac{\partial \mathbf{B}}{\partial t} \cdot d\mathbf{A} = -\oint \mathbf{E} \cdot d\mathbf{l} \qquad (2)$$

Now, let us apply this to the right-facing face on the parallelepiped centered at $i, j+\frac{1}{2}, k+\frac{1}{2}$. The discretized time derivative of the face normal component of **B**, B^1, is given by

$$\left[\frac{\left[\left(B^1\right)^{n+1}-\left(B^1\right)^n\right]}{\Delta t}\frac{\mathbf{l}_1}{l_1}\cdot\mathbf{A}^1\right]_{i,j+\frac{1}{2},k+\frac{1}{2}}$$

$$=-\left(\mathbf{E}^{n+\frac{1}{2}}\cdot\mathbf{l}_3\right)_{i,j+1,k+\frac{1}{2}}+\left(\mathbf{E}^{n+\frac{1}{2}}\cdot\mathbf{l}_3\right)_{i,j,k+\frac{1}{2}} \tag{3}$$

$$-\left(\mathbf{E}^{n+\frac{1}{2}}\cdot\mathbf{l}_2\right)_{i,j+\frac{1}{2},k}+\left(\mathbf{E}^{n+\frac{1}{2}}\cdot\mathbf{l}_2\right)_{i,j+\frac{1}{2},k+1}$$

where, the $\mathbf{l}_i$'s are the cell-edge tangent vectors, which reside at the center of cell edges; an example of which is given by

$$\left(\mathbf{l}_2\right)_{i,j+\frac{1}{2},k}=\mathbf{r}_{i,j+1,k}-\mathbf{r}_{i,j,k} \tag{4}$$

The $\mathbf{l}_1$ is obtained by differencing on the i-index, and $\mathbf{l}_3$ by differencing on the k-index. The same procedure can be carried out to evaluate the time derivative of $\mathbf{E}$ working on the E-cell. The calculation of the position vector $\mathbf{r}_{i,j,k}$ will be explained in Section 4 which describes the grid generation procedures used. The component B^1 is defined through

$$\mathbf{B}=\mathbf{e}_1B^1+\mathbf{e}_2B^2+\mathbf{e}_3B^3 \tag{5}$$

where $\mathbf{e}_i=\mathbf{l}_i/l_i$ .The component, say, B_1 can be obtained by taking the scalar product of the above expression with $\mathbf{l}_2\times\mathbf{l}_3$, which is the area of face 1.

The tangential electric field appearing in (3) is therefore given by

$$\mathbf{E}\cdot\mathbf{l}_i=\mathbf{l}_i\cdot\mathbf{e}_1E^1+\mathbf{l}_i\cdot\mathbf{e}_2E^2+\mathbf{l}_i\cdot\mathbf{e}_3E^3 \tag{6}$$

In an orthogonal coordinate system, only the E^i term would be present. Since the various components of $\mathbf{E}$ reside at different cell faces, second-order accuracy requires the other components be interpolated to the position occupied by $\mathbf{E}\cdot\mathbf{l}_i$. For example computation of $\mathbf{E}\cdot\mathbf{l}_1$ requires

$$\left(E^2\right)_{i+\frac{1}{2},j,k}=\tfrac{1}{4}[\left(E^2\right)_{i,j+\frac{1}{2},k}+\left(E^2\right)_{i,j-\frac{1}{2},k}+\left(E^2\right)_{i+1,j+\frac{1}{2},k}+\left(E^2\right)_{i+1,j-\frac{1}{2},k}] \tag{7}$$

and a similar interpolation is required for the E^3-component.

The above discussion indicates the steps necessary to evaluate the time derivative of the magnetic field from Faraday's law. The advancement of the electric field from Ampere's law follows a similar procedure.

2.2 The Particle Equations of Motion.

The standard relativistic equations of motion in the units of (1) are

$$\frac{d\mathbf{u}}{dt}=\mathbf{E}+\mathbf{u}\times\mathbf{B}/\gamma \tag{8a}$$

$$\frac{d\mathbf{x}}{dt}=\mathbf{u}/\gamma \tag{8b}$$

where $\mathbf{u}$ is the momentum per unit mass, and $\gamma=\sqrt{1+u^2/c^2}$. The time stepping of these equations is standard and will not be repeated here. Instead the discussion will focus on those aspects of integration of (8) peculiar to curvilinear coordinates. The main difficulty posed by use of curvilinear coordinates is that evaluation of the derivative in (8a) requires differentiation of the of the metric coefficients, the $\mathbf{l}_i$ in (4). It will be noticed that this has

been avoided in the evaluation of the *curl* operations. To avoid differencing errors, we choose to keep the components of **u** in Cartesian components. This requires conversion of **E** and **B** from face-normal to Cartesian components. The conversion simply requires resolution of the x, y and z-components of the $\mathbf{e}_i$ appearing in (5). The components are easily computed from x, y and z components of the coordinate positions appearing in (4). As previously indicated, the particle charge density resides at the center of the E-cell, $\mathbf{r}_{i,j,k}$. Momentum conservation requires that both the **E** and **B** fields be interpolated from the position at which the charge density resides. The code therefore interpolates the face-normal components to the E-cell center prior to computation of the Cartesian components.

Equation (8b) is integrated in curvilinear coordinates. The reason is that it is the curvilinear coordinate that serves as a pointer to the grid points surrounding the particle. In fact, the grid point indices are simply obtained by truncating the curvilinear position. PIC (particle-in-cell) weighting is used to interpolate the fields to the particle position. Updating the curvilinear position requires conversion of the velocity to face-normal components. The curvilinear velocity component, $\dot{q}^1$ can be obtained by taking the scalar product of the relation

$$\left(\mathbf{1}_x u_x + \mathbf{1}_y u_y + \mathbf{1}_z u_z\right) / \gamma = \mathbf{l}_1 \dot{q}^1 + \mathbf{l}_2 \dot{q}^2 + \mathbf{l}_3 \dot{q}^3 \qquad (9)$$

with $\mathbf{l}_2 \times \mathbf{l}_3$ This generates a 3×3 matrix transformation. The curvilinear velocity is now position dependent. The transformation elements reside at the center of the E-cells, and the transformation is evaluated at the particle by PIC interpolation. To preserve second order accuracy, the particle's curvilinear position is advanced a half time step. The transformation evaluated at the half time step position is then used to advance the particle's curvilinear position a full time step. This procedure makes the curvilinear move a rather computationally expensive part of the particle update.

2.3 Calculation of the Current.

This subsection describes the implementation of the Villasenor-Buneman [1992] exactly charge-conserving algorithm for calculating the current. The essence of the algorithm is illustrated in Figure 4, which shows the finite element of charge carried by a particle on a rectangular grid. The particle element is the same size as the grid cell. The current is the amount of charge, represented by the change in area, carried across a cell boundary by a particle in a time step, divided by the duration of the time step. In two dimensions the current vector resides on the center of the cell boundary and is normal to the boundary. Since the amount of flux crossing a boundary represents the decrease (increase) in area occupied by the particle within a given cell, the current is rigorously conservative. In three dimensions the components of the current are centered on the cell face and normal to the cell face.

In curvilinear space, the coordinate system is a cubic lattice with unit distance between grid points. Hence the conversion to curvilinear is trivial. The current comes out automatically to have face-normal components centered on faces of the E-cells – just where it is needed to update the E-field from Ampere's law. The particle's motion in curvilinear space is computed with the $\dot{q}^l$ that is needed to update the particle position.

A particle's charge element can intersect a rather large number of cell faces during a time step. Villasenor and Buneman use a rather complicated decision tree to calculate the amount of charge crossing each cell face. We have implemented a conceptually simpler recursion process. In our code the current calculation is done in conjunction with the particle move. The move is stopped each time a particle's finite element crosses a new cell face. The time is recorded and a new move is begun for the particle starting at the recorded time. The particle can recirculate through the move loop an arbitrary number of times until the duration of the time step has elapsed.

3. Multiple Coordinate Patches

The major accomplishment of this project has been the extension of the methods described above to coverage of a domain by an arbitrary number of coordinate patches. The design philosophy has been to segregate the code. One set of subroutines operates on coordinate patches independently, and another set of subroutines takes care of passing data between coordinate patches. This allows direct incorporation of methods already developed for curvilinear coordinates into the multiple coordinate patch environment.

This has required some change in the structure of the data. Flexibility and generality require that the code be allowed to loop through an arbitrary number of patches, each of which may be dimensioned differently. This flexible data structure is accommodated in the Fortran 90 language used in the project. In this language, a component of the electric field may be referenced as

```
f(m)%e(i,j,k,n) = ...
```

where `f` is a *type*, or *structure* in *C* parlance, describing the field variables. Other members of the type are the electric field and current density. The *m* index is the patch number. The *i, j, k* indices refer to the grid point, and are used in the same context as the in Section 2. The index *n*, which runs from 1 to 3, is the component number. The dimension declaration on the positions of *i, j, k* indices depend on the patch number.

Particles comprise a different data structure. A particle's curvilinear position may be referenced by

```
p(m)%q(k,n) = ...
```

where `p` is the *type* describing the particle arrays, *m* is the patch index , *k* is a particle index, and *n* is the component of the particle's position vector. All of the particles corresponding to patch number *m* reside within the volume occupied by coordinate patch *m*. The *k*-slot is dimensioned sufficiently large to accommodate all the particles likely to be within the volume of patch *m*. The particle arrays are allocable, so it is possible to re-dimension them from time to time, as particles move from one region to another.

The code is written for the Cray Y-MP, but the structure can be efficiently implemented on a distributed memory parallel computer by mapping patches onto individual processor elements. The sharing of data between coordinate patches minimizes inter-patch/processor communication by keeping the bulk of data references local to a patch/processor.

3.1 Sharing of Field Data

The key to modularization of the computation is data sharing. Figure 5 shows a blow-up of the grid shown in Figure 1 around the region surrounding the intersection of three coordinate patches. Also shown is the dual grid structure. The solid lines denote the

B-cells, while the dotted lines are the E-cells. The density resides at the intersection of solid lines. The dashed grid is displaced a half grid distance out of the page from the solid grid. Data on the heavy grids is common to two or more patches. The choice of location of the density, and hence the E and B-cells is dictated by compatibility with the Villasenor-Buneman current algorithm across patch boundaries.

Consider the interface between patches 1 and 5 and the update of B^3 and E^3, which are field components normal to the page. The B^3 resides on the intersection of dashed lines, and is updated by taking the line integral of **E** on the surrounding solid lines. The update occurs in patch 5. The update of E^3 on the other hand occurs in patch 1. Because the data is shared, the update is done entirely within a patch. Once the update on each of the patches is complete, data on the solid boundary line must be shared with patch 5, and data on the heavy dashed line must be copied from patch 5 to patch 1.

The data sharing is accomplished through separate interface subroutines for the **B** and **E** fields. The subroutines are invoked twice: Once operating on the face normal components following the update of the fields, and once again on the tangential components of the fields following the conversion of the field from face normal to edge tangential components. The interface routines loop through all of the patches. Within each patch loop are nested loops over the three coordinats and each of the six faces. The controlling parameters determine whether the patch boundary is an external boundary or an inter-patch boundary. In the case of an external boundary, the interface routine sets the appropriate components to the values required by the boundary condition. In the case of an inter-patch boundary, the number of the adjacent patch is identified.

The interface routine then, where appropriate, assigns values of the field variables from the adjacent patch to its grid. On the interface between patches 1 and 5 in Figure 5, the 1-components of both patches increase to the right and the 2-components both increase in the upward direction. Thus the interface routine operating on patch 5 simply copies values B^1,E^2 and E^3 lying on the solid line to its grid. When it operates on patch 1, values of B^2, B^3 and E^1 from patch 5 are copied to patch 1. In the case of the interface between patches 2 and 5, the 1-coordinate increases upward and the 2-coordinate increases to the left. This means that B^1 copied from patch 2 becomes B^2 when transferred to patch 5 , and E^2 becomes $-E^1$ when transferred to patch 5. The reverse occurs when variables on the heavy dashed line are transferred from patch 5 to patch 2. Also, since grid points are sometimes enumerated in opposite directions, the interface routine must take care of relabeling of grid indices. There are in fact six parameters that identify adjacent patches and are responsible for relabeling and re-numerating of grid points for every patch component on every face of each patch.

3.2 Coordinate Discontinuities

The above procedure of copying and relabeling of field components is appropriate to boundary transitions between coordinate patches that are continuous. Figure 6 illustrates the difficulties when the methods described above are applied to discontinuities between coordinate patches. One problem is illustrated in the case of E^1. The face normal component that emerges from taking the *curl* of **B** is displaced the cell-edge component. In the case of the 1-component of the magnetic field, the center of a coordinate face is displaced from the center of a coordinate edge. Moreover, the coordinate directions

change so drastically that the face normal direction one side of a cell bears little resemblance to the face normal direction on the other. The usual interpolation methods described in the previous section will fail to give an accurate value for the tangential component, B_1. Since the *curl* operation takes differences between field components, acceptable accuracy requires very accurate values for the tangential components.

Since the interpolation methods described in the previous section will not work in the neighborhood of a coordinate discontinuity, more general methods for interpolating in both direction and position are needed. This can be done by using the expansion

$$\begin{aligned}\mathbf{E} = \mathbf{1}_x\left[a_x + b_x(x-x_0) + c_x(y-y_0) + d_x(z-z_0)\right] + \\ \mathbf{1}_y\left[a_y + b_y(x-x_0) + c_y(y-y_0) + d_y(z-z_0)\right] + \\ \mathbf{1}_z\left[a_z + b_z(x-x_0) + c_z(y-y_0) + d_z(z-z_0)\right]\end{aligned} \tag{10}$$

The tangential component of, say, E_1 at the point x_0, y_0, z_0. is calculated by taking the scalar product of (10) with $\mathbf{l}_1$, defined in (4). The result is

$$E_1 = l_{1x}a_x + l_{1y}a_y + l_{1z}a_z \tag{11}$$

To determine the Cartesian components of **a**, **b**, **c** and **d** in (10), we equate the expression with

$$\mathbf{E} = \mathbf{e}_1E^1 + \mathbf{e}_2E^2 + \mathbf{e}_3E^3 \tag{12}$$

at twelve neighboring points where the face normal E^i are known. For example, say we know E^2 at the point x_j, y_j, z_j, then one of the twelve relations needed to determine the coefficients in (10) is

$$\begin{aligned}\mathbf{e}_2\cdot(\mathbf{e}_3\times\mathbf{e}_1)E^2 = \mathbf{1}_x\cdot(\mathbf{e}_3\times\mathbf{e}_1)\left[a_x + b_x(x_j-x_0) + c_x(y_j-y_0) + d_x(z_j-z_0)\right] + \\ \mathbf{1}_y\cdot(\mathbf{e}_3\times\mathbf{e}_1)\left[a_y + b_y(x_j-x_0) + c_y(y_j-y_0) + d_y(z_j-z_0)\right] + \\ \mathbf{1}_z\cdot(\mathbf{e}_3\times\mathbf{e}_1)\left[a_z + b_z(x_j-x_0) + c_z(y_j-y_0) + d_z(z_j-z_0)\right]\end{aligned} \tag{13}$$

The set of twelve equations generated are then solved to determine the coefficients in **a**. Given the *template* of twelve face center points it is possible to compute any component of the **E** or **B** fields, say E_i

$$E_i = \sum_{n=1}^{12} a_{in}E^n \tag{14}$$

where the a_{in} are precomputed by taking the inverse of a 12×12 *template matrix*.

The same procedure is applied to the magnetic field, **B**. In addition it is necessary to determine E^1 on each side of the discontinuity, as illustrated in Figure 6. This is done by taking the scalar product of (10) for **E** with $\mathbf{e}_2' \times \mathbf{e}_3$, where $\mathbf{e}_2'$ is the unit vector in the direction of the lighter-faced dashed lines shown in Figure 6.

Figures 7a, b show the *target* points and component directions where the B and E-fields were computed. The components on and just to either side of the diagonal were computed for all 17 points out along the diagonal. The same computation is performed for all points along the axis normal to the plane of the page. The computations are repeated for the interface of patches 2 and 3, 3 and 4, and 4 and 1. The source components, E^n, in (14) were generally chosen so that there would be four values of the 1-component, four of

the 2-component and four of the 3-component. The locations from which they were each selected formed the points of a tetrahedron that contained the target point. The reason for this is that if any of the face normals in the same direction were on a plane, the template matrix would be singular.

For target points for B_2 and E_1 located along the diagonal, the coordinate system shows considerable symmetry, which cannot be reflected in the tetrahedral arrangement of source points. For this reason, a thirteenth term and coefficient is added to the expression (10). This term is quadratic in a variable that increases in a direction perpendicular to the heavy diagonal line shown in Figure 5 and is in a direction perpendicular to the plane of the discontinuity. Caution has to be exercised in situations in which a target point is located on a symmetry axis because the upper left-hand block of the template matrix could be singular, even though the entire template matrix is not. This could cause failure or large truncation errors in the routine that inverts the template matrix. This can be avoided by randomizing the enumeration of the source points.

The calculation of the Cartesian components needed for boosting the particles for the most part can be done using the methods described in Section 2.2. This procedure does become inaccurate points on the chevron discontinuity. There the expression in (10) is invoked to calculate the Cartesian components.

The implementation procedures described above are outlined in Figure 8, which shows a block diagram of one-time setup and the routines that are called every time step. The setup routine is called by the subroutine that computes all the curvilinear coefficients relating to the patch interior.

The setup routine contains a set of numbers that identify all the target points and their components for each of the peripheral coordinate patches. Attached to each target point is a set of integers that identify the location of source, or template, points and their associated components. Since many of the target points share the same source points, a subroutine was written to compute the x, y, z components of $(\mathbf{e}_1 \times \mathbf{e}_2)$, $(\mathbf{e}_2 \times \mathbf{e}_3)$, or $(\mathbf{e}_3 \times \mathbf{e}_1)$ for each of the source points, as indicated in (13). The subroutine call takes as an input the indices denoting position and component of the source points. The same routine also computes the x, y, z components of the edge tangent vector, since many of the source point are also target points. In addition the routine also computes both the face center and edge center positions. The next step in the setup routine consists of a loop over target points. This loop calls a procedure which computes the template matrix, inverts it and finally computes the coefficients a_{in} in (14).

In the main time stepping loop in which the fields are updated, the calculation of the target components is done in the interface routine. We start from the face-center components, as output by the *curl* operation. The first step is to execute the data share procedure described in section 3.1, even for components on the discontinuous patch interfaces. Next, the source components are saved in arrays that can be addressed by the indices declared in the setup routine. Then in the case of **E**-field, two face normal components of E^1 are computed for each side of the chevron discontinuity, using coefficients generated by the setup procedure. The interface procedure is then exited.

The interface procedure is entered again following the procedure that computes the edge-tangent components. The source components that were saved in the previous call are

then used to compute the edge-tangent components in the neighborhood of the coordinate discontinuities, making use of the a_{in} coefficients computed in the setup procedure. Following this, the interface procedure described in Section 3.1 is again executed. The fields are then ready for the *curl* computation.

3.3 Testing and Evaluation

The primary test of the algorithms described in this section was to take the *curl* of vector fields which changed linearly with distance, i.e. $\mathbf{B} = \mathbf{1}_y z$, which yields a unit vector in the x-direction. The tests were based on use of the coordinate system depicted in Figures 1 and 2. The steps used in the test are : (1) The Cartesian vectors were converted to the curvilinear, face-centered components that would be found following the *curl* operation. (2) The interface routine was applied, as described above. (3) Then the face-centered components were converted to edge-tangent components. (4) The interface routine was then applied. (5) The *curl* was then taken, and (6) the result was converted back to Cartesian components for comparison with the known result.

The interface routines were first applied without the procedures described in Section 3.2. Errors were unacceptably large around the target points shown in Figures 7a, b. Attempts to improve the accuracy were made by smoothing out the chevron discontinuity between the peripheral patches. The result was an increase in the number of grid points with unacceptably large (100%) errors. Moreover, no rearrangement of coordinate points improved the accuracy around the degenerate cells at the interface of three coordinate patches.

The test procedures using linearly varying Cartesian components made it easy to compute edge-tangent field components by hand. It was found that the errors could be almost entirely accounted for by errors in the edge tangent components. Since the *curl* is computed by differencing edge-tangent components, they had to be computed with a high degree of accuracy. An error of a couple percent could give a 100% error in the result of the *curl* operation. Errors for the *curl* operation have been generally brought below 10% for the grid shown in Figures 1 and 2 and the target points shown in Figures 7 a, b. The errors could be reduced further by increasing the number of target points, because the errors remaining are contributed by edge-tangent components that were not computed by the methods described in Section 3.2.

As a final remark, the procedures described above are sufficiently flexible and modular that they could be applied to the entire grid without a great deal more coding. It would just require some expansion in the tables that specify the template of source points. However, far enough away from the grid discontinuities, the errors are considerably less than 1%. Moreover, the source point template being based on a tetrahedral configuration may not have the accuracy of a more centered interpolation methods in the interior of coordinate patches, although it would work well for linearly varying fields used in the tests. Another distinction is that the tetrahedral interpolation relies on twelve or more source points and twelve or more coefficients, while the centered interpolation requires nine source points and nine coefficients.

The selection of source and target points relied to some extent on the fact that coordinate scale factors varied relatively slowly along the cylindrical axis. However, the general methods are independent of the orientation of the underlying x,y,z Cartesian

coordinates. Thus the methods described in Section 3.2 can be applied irrespective of the orientation of either the curvilinear or Cartesian coordinates to the coordinate discontinuity. They could in fact be applied to any finite element mesh, given an appropriate template of source points.

3.4 The Particle Mover and Current Algorithm

The presence of patch boundaries requires a rather straightforward extension to the particle move/current algorithm described in Section 2. A particle is tagged by a patch number and an index number. In addition to the particle's phase space position, the particle carries a logical variable. When it is `.TRUE.` the particle is moved and boosted, and when it is `.FALSE.` it is ignored. When a particle moves to a new patch, the time step is recorded the logical variable is set to `.FALSE.`, and the particle is placed in a buffer along with the time and the index of the destination patch. An interface routine then places the particle at the top of the stack in the destination patch, assigns an index and sets the logical variable to `.TRUE.` The move subroutine is entered again, and the new particles are moved. The few particles which cross into a new patch are again placed in a buffer, and the process is continued. The move process stops when the buffer is empty. At the end of the moves, the particle arrays have a number of empty slots. A stack push down is executed in each patch so that all active particles are placed consecutively in the particle data arrays for each patch.

A particle that is in the process of crossing a patch boundary contributes to the particle flux from both patches to the common boundary face. The cell-face currents are accumulated during the particle move. A current interface routine sums these currents and then distributes the data in same the way that the E-field data is distributed.

4. Grid Generation

The grids shown in Figures 1 and 2 were generated by two-dimensional solutions to the Laplace equation. Figures 9 and 10 show contour plots of the potential functions. The potential in Figure 9a was used to generate the nearly rectangular coordinates in the inner patch to the "radial" coordinates in the peripheral patch. Figure 9b shows contours which change continuously from a straight-line boundary to a circular boundary in the peripheral patches, while Figure 10a shows the potential function which conforms to the axial shape of the domain.

Coordinate generation involves two steps. The first is the generation of the potentials, and the second step involves the conversion of the potential functions to the x, y, z location of coordinate points that can be used by the simulation program.

4.1 Generation of Potentials

This subsection describes the numerical technique for solution of Laplace's equation that satisfies Dirichlet conditions on bounding curves of arbitrary, but continuous, shape. The starting point is a Greens function solution that satisfies

$$\nabla^2 G(\mathbf{x};\mathbf{x}') = -\delta(\mathbf{x}') \tag{15}$$

i.e the potential of a unit source at an arbitrary location. This is done on an $m \times n$ grid. The solution satisfies periodic boundary conditions in one direction and force-free boundary condition in the other direction. It is obtained by use of an FFT in one direction and tri-

diagonal methods in the other direction. The solution possesses translational invariance and also has the property $G(\mathbf{x},\mathbf{x}') = G(\mathbf{x}',\mathbf{x})$. That is, the potential contours are the same irrespective of the location of the source point relative to the boundary. Therefore only one Greens function is needed.

The solution to the Laplace equation can be written as a linear combination of an arbitrary number of Greens functions

$$\Phi(\mathbf{x}) = \sum_{k=1}^{N} \alpha_k G(\mathbf{x},\mathbf{x}_k) \tag{16}$$

Say, we want Φ to equal B_j at points $\mathbf{x}_j$, $j = 1 .. N$. Then we get the relation

$$B_j = \sum_{k=1}^{N} \alpha_k G(\mathbf{x}_j,\mathbf{x}_k) \tag{17}$$

If the points $\mathbf{x}_j$ constitute the points defining the boundary, then (16) represents the solution that satisfies the boundary conditions with α_k determined by solution of the linear set of equations (17)

In order that the solution be smooth, the boundary points should be no more than one grid point apart. The potentials shown in Figures 9 and 10 were generated typically on a 256×260 grid, and about 800 points were used to define the boundary. This large a set of equations with source points so close together becomes ill-conditioned. To get around this problem 8 or 10 groups of equations, involving 100 or 80 points are solved separately and solutions to the various groups are averaged. The points in each of the groups are staggered such that if the set of points is broken into 8 groups, then each group consists of every tenth point. This does result in some loss of resolution. If there is some point where it is important that the solution accurately satisfy a particular value, then that point can be represented in each of the groups.

4.2 Determination of Grid Point Positions

The grid shown in Figure 1 was generated with two potentials. The starting point is to lay out a set of fiducial points of q^1 along a known curve. In the case of the center patch of the grid, the points are laid out on the horizontal straight line. This constitutes one point on the q^1 coordinate. Then a set of fiducial values of q^2 are established. In the case of the center patch these are the same values established for q^1 coordinate. The table for the potential q^1 contoured in Figure 9a is numerically differentiated to obtain $\partial q^1/\partial x$, $\partial q^1/\partial y$. These derivatives are used to determine the line of constant q^1 in x-y coordinates. The generation program calculates the curve of constant q^1 , $x(s)$, $y(s)$. In the course of tracing this line, the program evaluates $q^2(x,y)$. If a fiducial value of q^2 is exceeded, the program interpolates between the present q^2 and the value of q^2 at the previous integration step to find the value of s that corresponds to the fiducial value of q^2. The program then determines the values of x, y that corresponds to the fiducial value of q^2. The value of q^1 was set at the start of the curve tracing procedure. The x, y positions for fiducial value of q^2 are then recorded. The index for q^2 is incremented and the program seeks the crossing of the next fiducial value of q^2. The same type of procedure is used to fill out the coordinate points in the peripheral patches. This time however, the program traces out radial lines using the potential in Figure 9a and calculates the crossings of fiducial values of the potential shown in Figure 9b.

The potential in Figure 10a is used to evaluate the q^3 coordinate. Here the procedure is somewhat different in that the program traces along a gradient, rather than along a line of constant potential. The program lays out a set of fiducial values for the q^3 coordinate along the axis of the cylindrical figure shown in Figure 2, which in this case happens to be proportional to the Cartesian z-coordinate. The program also identifies a set of fiducial values of the potential function shown in Figure 10a. It then integrates out along the gradient of the potential function in Figure 10a. This generates q^3 and q^f coordinates, where q^3 advances along the cylindrical axis, and lines of constant q^f are the potential contours of Figure 10a. The result is shown in Figure 10b.

The last step in the construction of the three-dimensional coordinate system is the rotation of the grid in Figure 10 about the cylindrical axis at the bottom and the mapping of the coordinate system shown in Figure 1 onto constant q^3 surfaces of revolution.

The above procedure generates coordinate points that are connected by the lines shown in Figure 1 and the solid lines in Figure 5. This generates the E-cells. The dual grid points are placed at the center of the E-cells, and the B-cells, indicated by the dashed lines on Figure 5, are formed by planes connecting E-cell centers.

References

Birdsall, C.K. and A. B. Langdon, *Plasma Physics Via Computer Simulation*, p351, McGraw hill, new York, 985

Eastwood, J.W., W. Arter, N.J. Brealey and R.W.Hockney, Body-fitted electromagnetic PIC software for use on parallel computers, *Computer Physics Communications, 87*, 155, 1995

Madsen, N.K., Divergence preserving discrete surface integral methods for Maxwell's curl equations using non-orthogonal unstructured grids, *J. Comp. Phys., 119*, 34, 1995

Swift, D. W., Use of a hybrid code for global-scale plasma simulation, *J. Comp. Phys, 125*, 109, 1996

Villasenor, J. And O. Buneman, Rigorous charge conservation for local electromagnetic field solvers, *J. Computer Physics Comm., 62*, 306, 1992

Figure Captions

Figure 1. The filling of a circular cross section of a cylinder with five coordinate patches. Chevron type coordinate discontinuities extend from the corners of the central patch along the diagonals.

Figure 2. Perspective view of a coordinate system in the interior of a cylinder of arbitrary cross section. The grid spanning the cross section is that shown in Figure 1.

Figure 3. The location of the face normal components of **E** and **B** on the *E*- and *B*-cells.

Figure 4. Illustration of the finite element of charge carried by a particle.

Figure 5. A detailed blow-up of the grid around the intersection of three coordinate patches. The charge density is at the intersection of the dashed lines, while the divergence of **E** is at the intersection of solid lines. Data on the heavy coordinate lines is shared in common between two or three patches.

Figure 6. Illustration of the tangential and face normal components near a chevron coordinate discontinuity. The superscripts denote face normal components, while subscripts denote edge tangent components.

Figure 7a. The locations where the tangential components of the **E** -field were computed by the special methods described in section 3.2 The arrows show the component directions.

Figure 7b. The locations where the components of **B** were computed by the special methods outlined in section 3.2.

Figure 8 A block diagram showing the relationship among the various modules used in updating the fields.

Figure 9. Potentials used in generation of the grid shown in Figure 1. Panel *A* shows the potential used in the generation of the grid of patch 5 and the radial lines in patches 1 - 4. Panel *B* shows the potential used to transition the grid from a square to a circle in patches 1 - 4.

Figure 10. Panel *A* : The potential used in forming the axial dependence of the grid. Panel *B*: The grid generated by that potential.

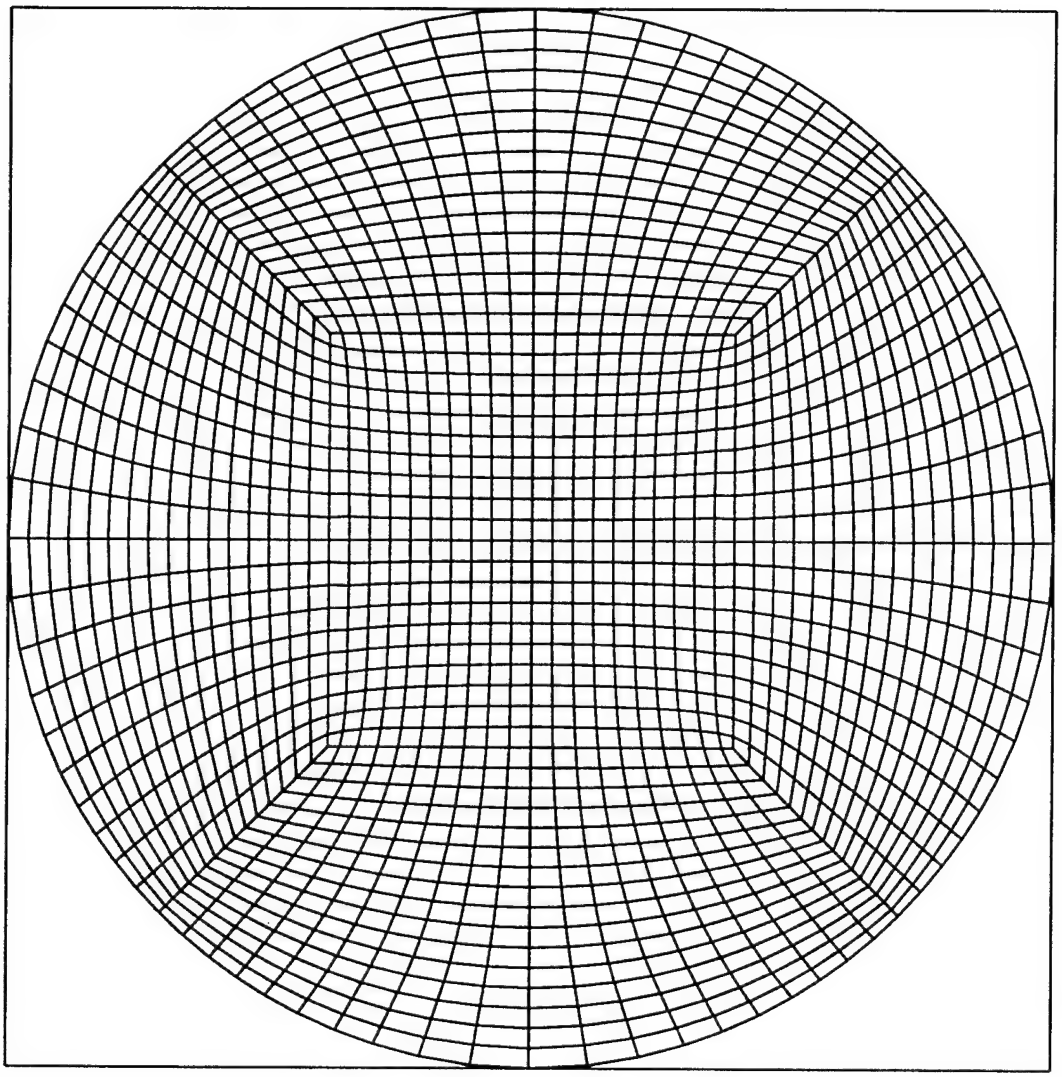


Figure 1 The filling of a circular cross section of a cylinder with five coordinate patches. Chevron type coordinate discontinuities extend from the corners of the central patch along the diagonals.

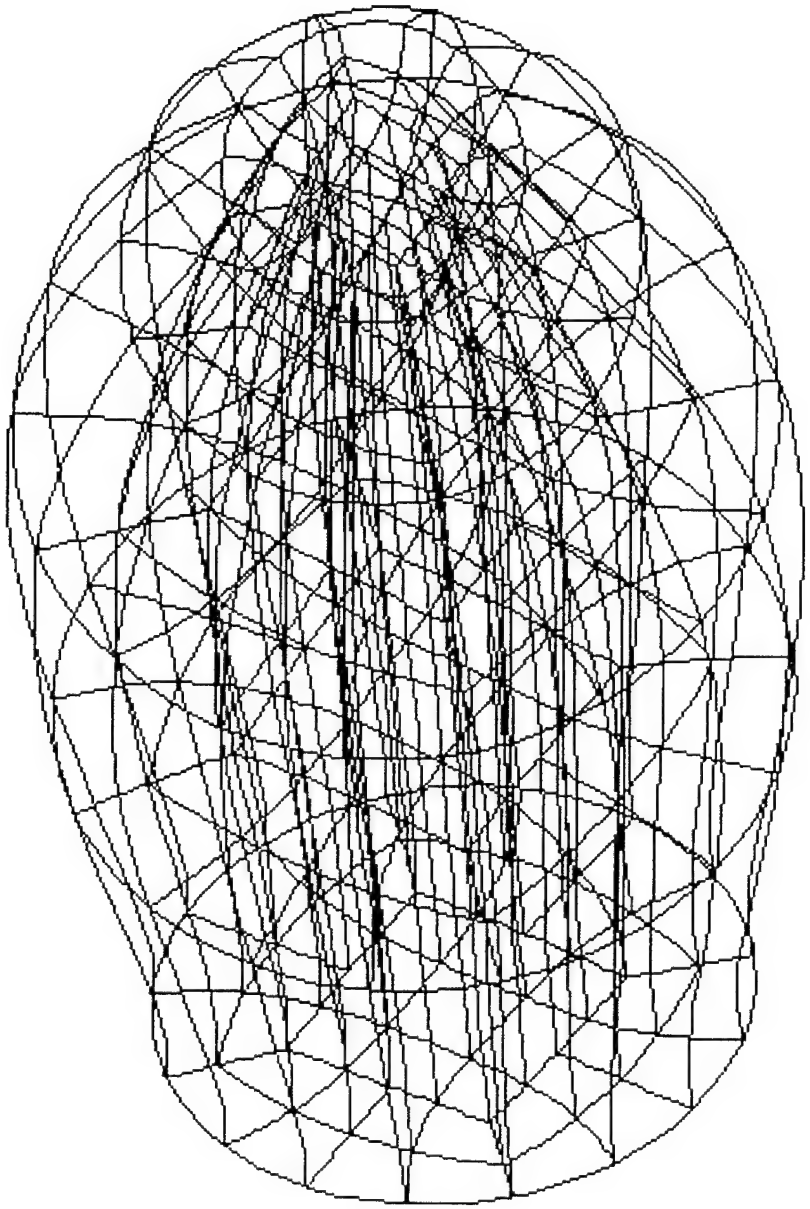


Figure 2 Perspective view of a coordinate system in the interior of a cylinder of arbitrary cross section. The grid spanning the cross section is that shown in Figure 1.

$$\left(E^1\right)_{i+\frac{1}{2},j,k}, \quad \left(B^1\right)_{i,j+\frac{1}{2},k+\frac{1}{2}}$$

$$\left(E^2\right)_{i,j+\frac{1}{2},k}, \quad \left(B^2\right)_{i+\frac{1}{2},j,k+\frac{1}{2}}$$

$$\left(E^3\right)_{i,j,k+\frac{1}{2}}, \quad \left(B^3\right)_{i+\frac{1}{2},j+\frac{1}{2},k}$$

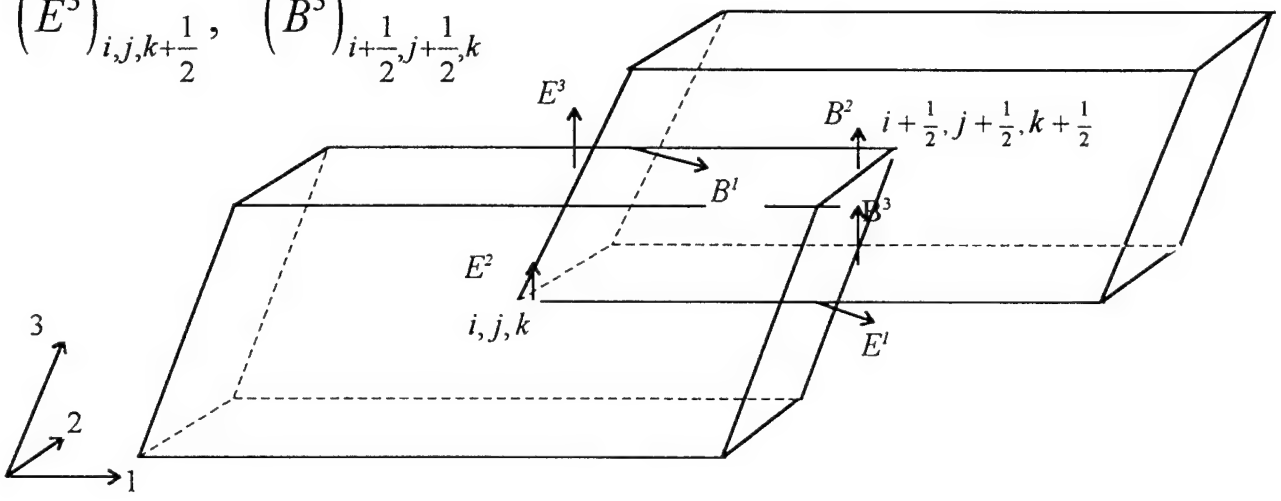


Figure 3 The location of the face normal components of **E** and **B** on the E- and B-cells

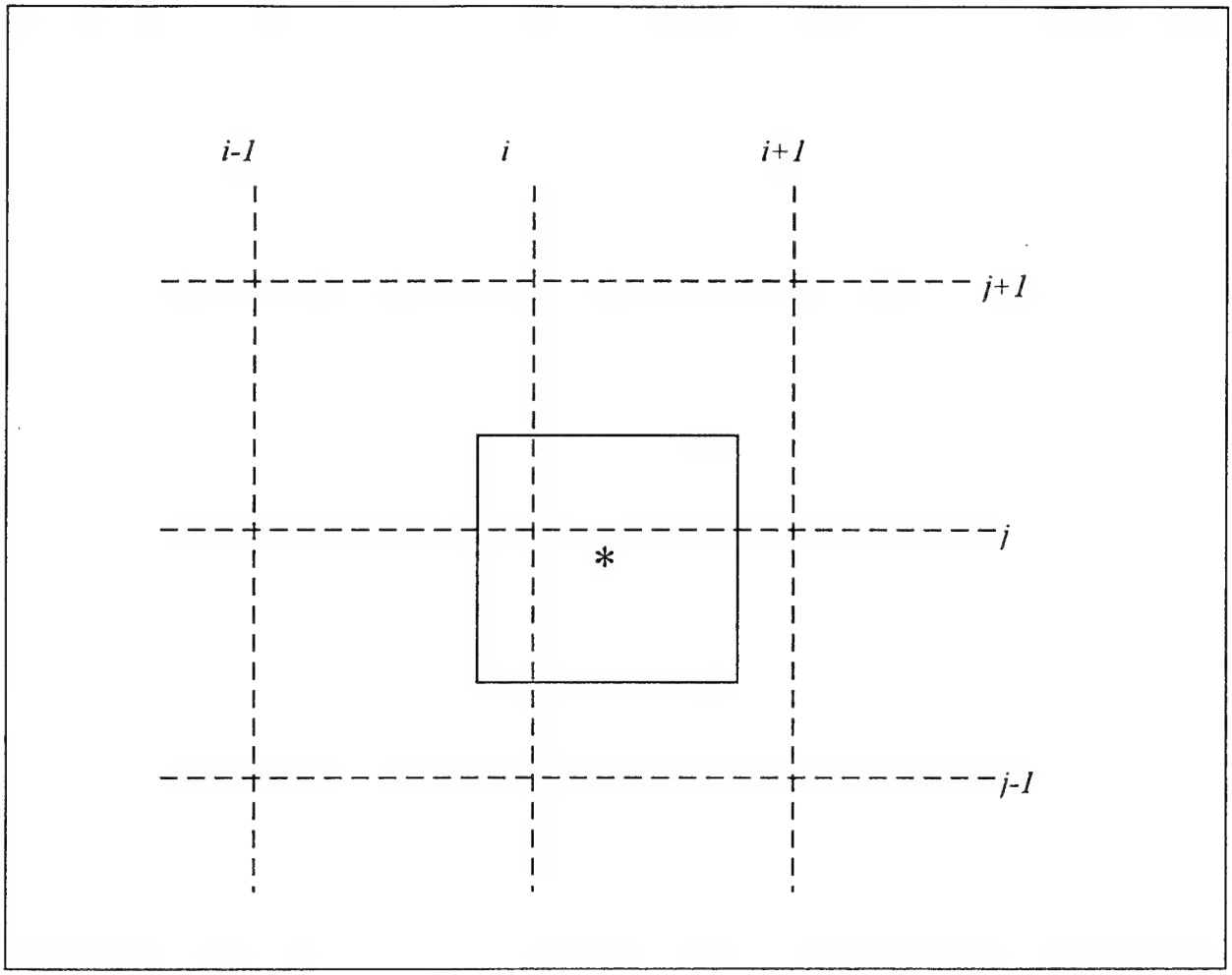


Figure 4. Illustration of the finite element of charge carried by a particle

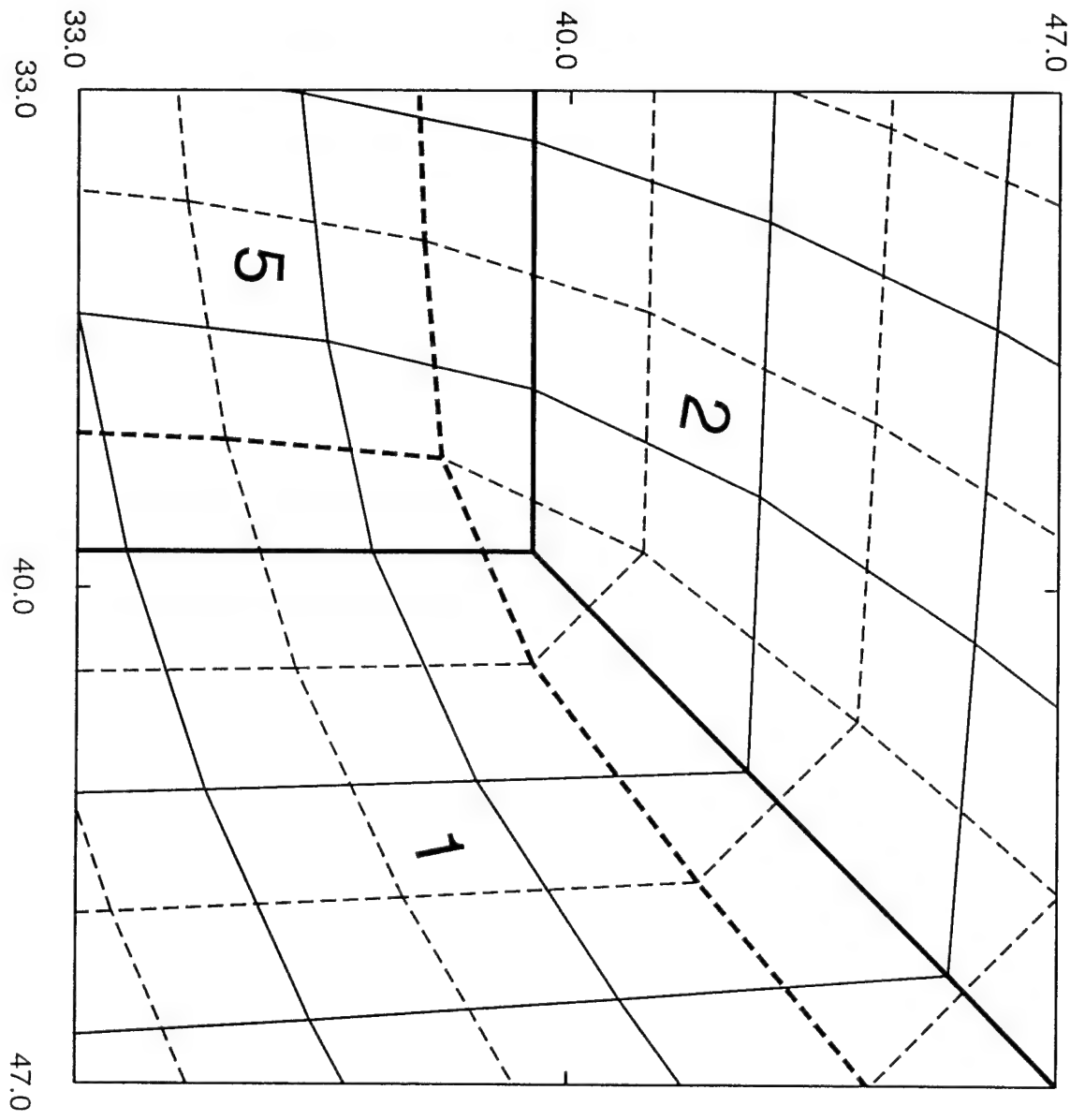


Figure 5 A detailed blow-up of the grid around the intersection of three coordinate patches. The charge density is at the intersection of the dashed lines, while the divergence of **B** is at the intersection of solid lines. Data on the heavy coordinate lines is shared in common between two or three patches.

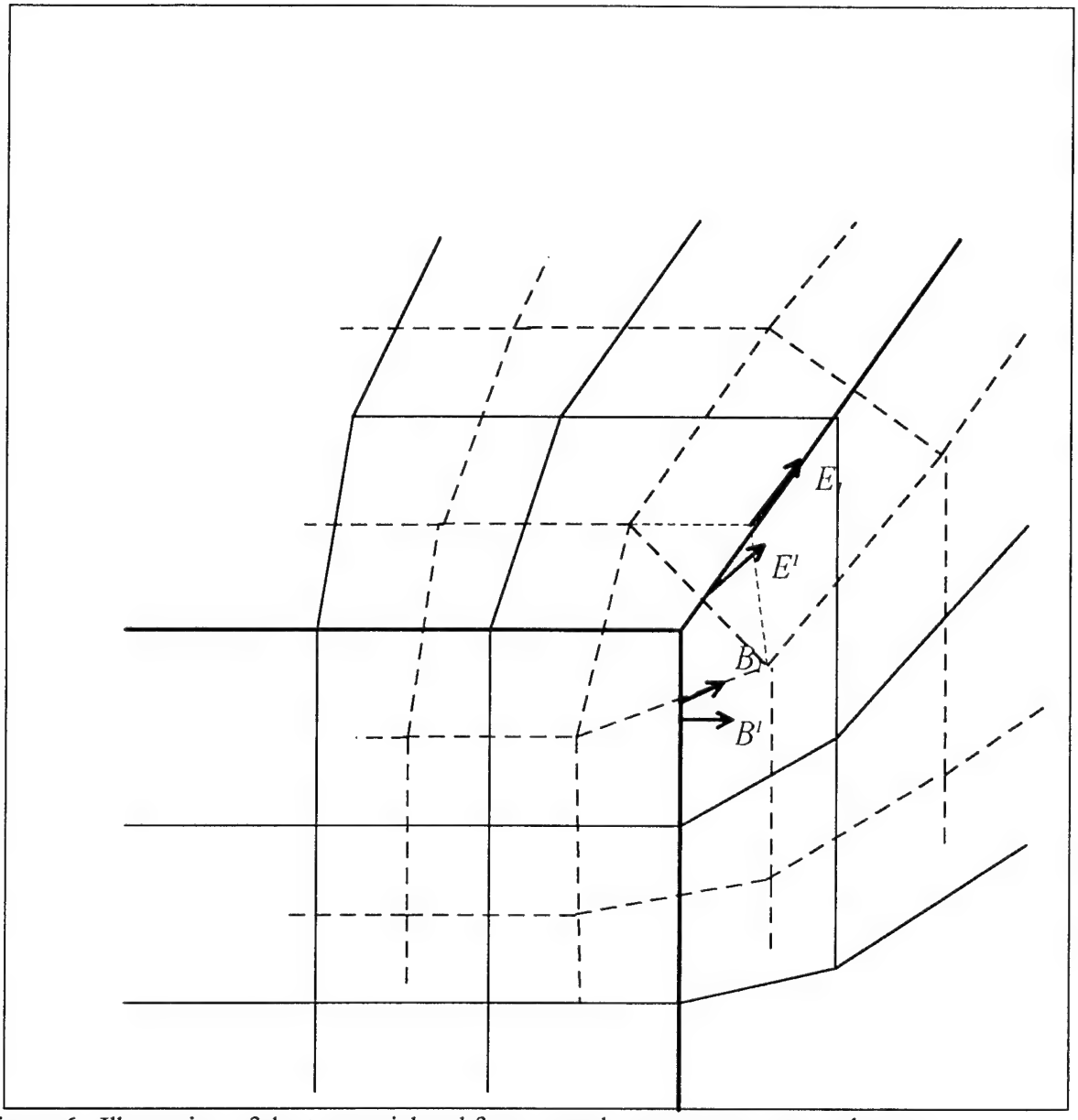


Figure 6. Illustration of the tangential and face normal components near a chevron coordinate discontinuity. The superscripts denote face normal components, while subcripts denote edge tangent components.

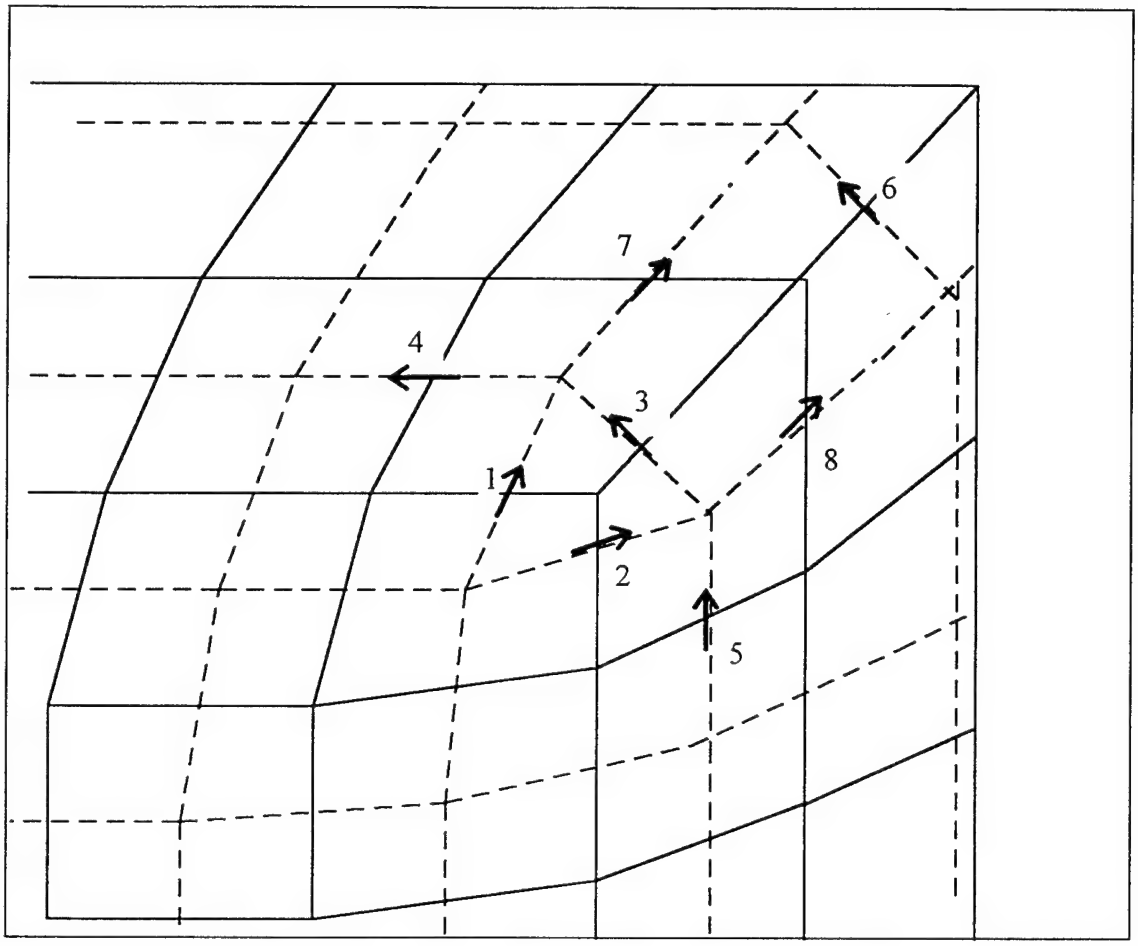


Figure 7a E-field target points. The arrows show component directions.

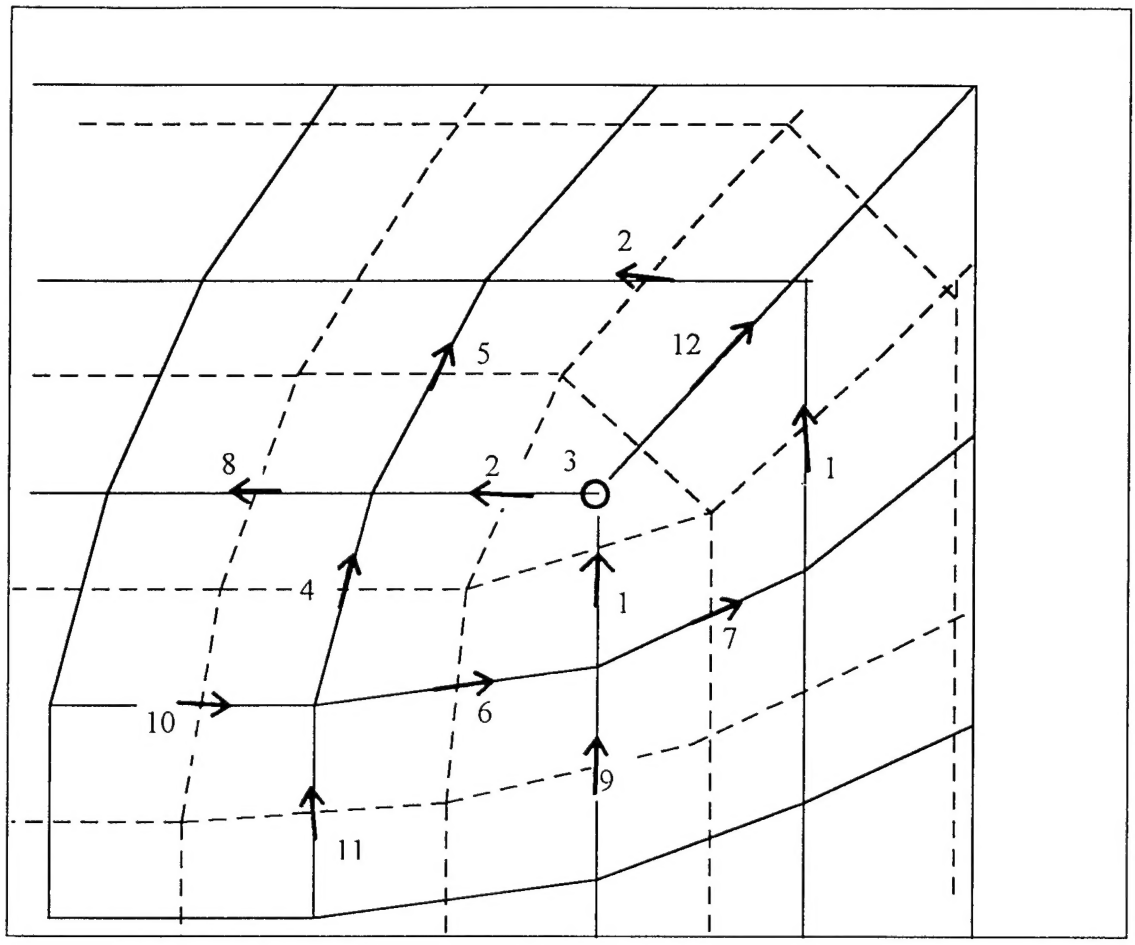


Figure 7b. *E*-field target points. The arrows show component directions

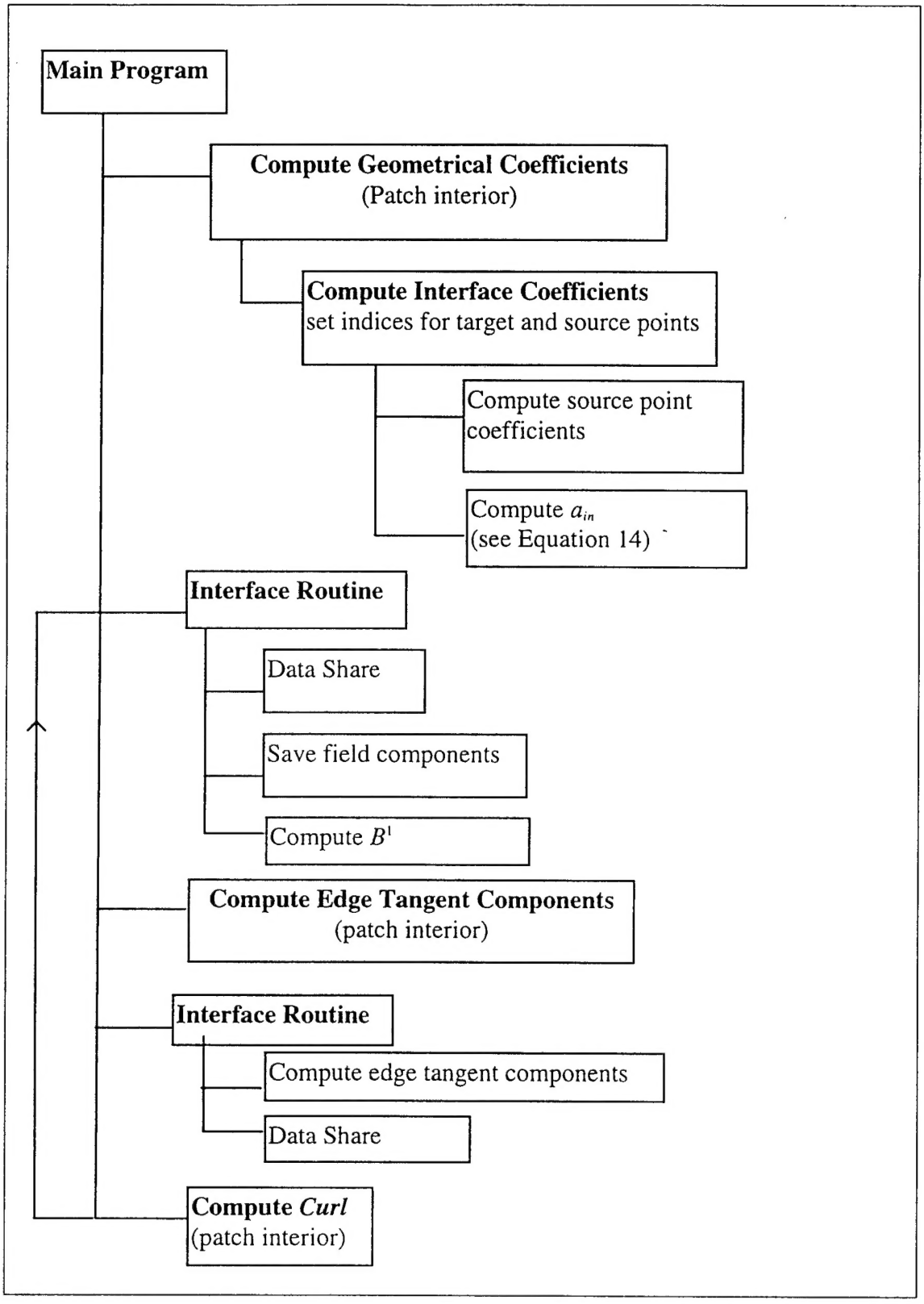


Figure 8 A block diagram showing the relationship among the various modules used in updating the fields.

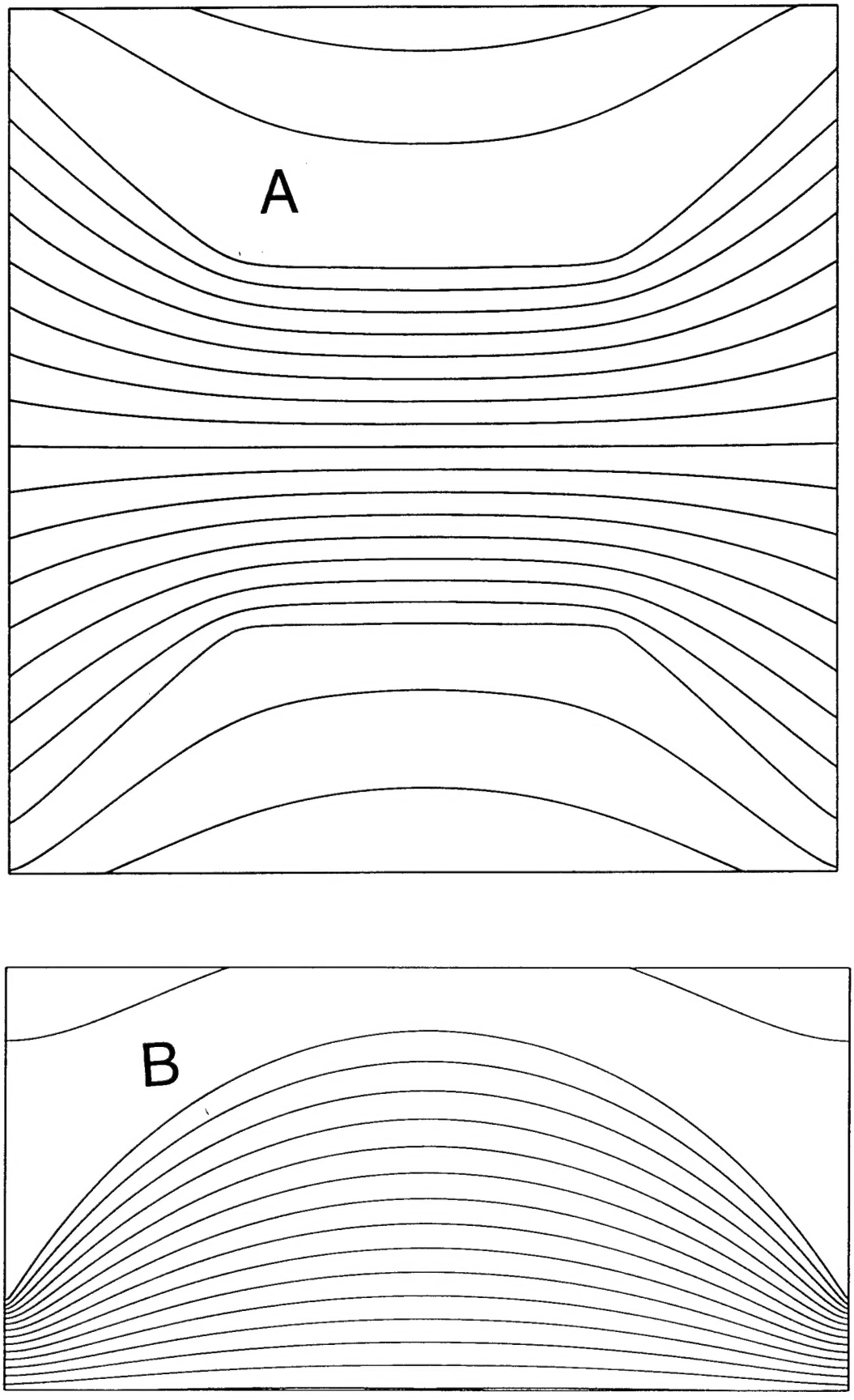
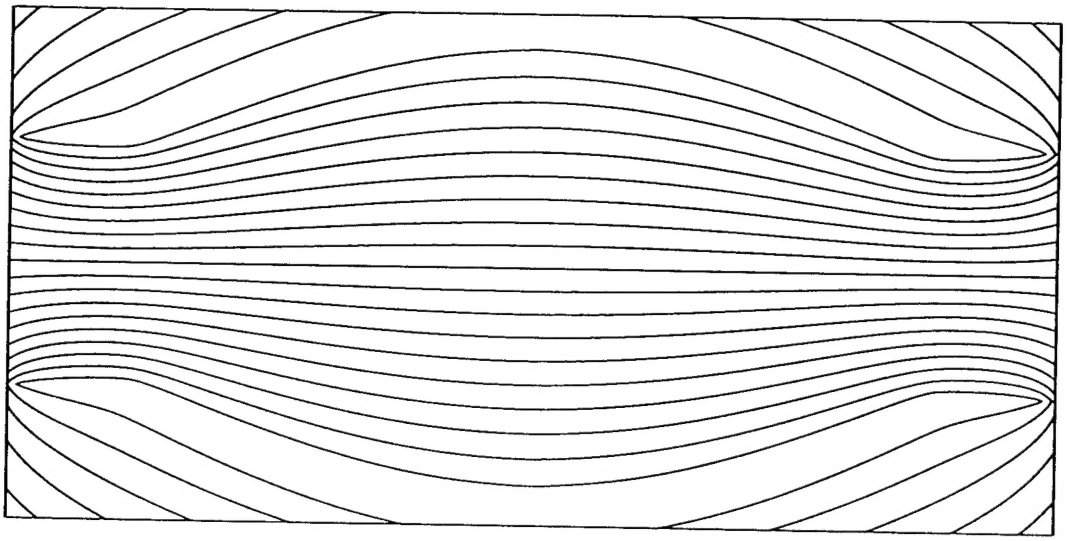
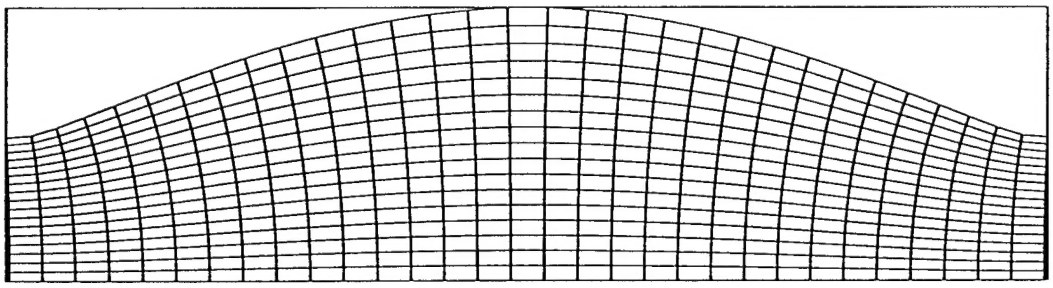


Figure 9 Potentials used in generation of the grid shown in Figure. Panel *a* shows the potential used in the generation of the grid of patch 5 and the radial lines in patches 1 - 4. Panel *b* shows the potential used to transition the grid from a square to a circle in patches 1 - 4.



A



B

Figure 10. Panel *a* : The potential used in forming the axial dependence of the grid. Panel *b*: The grid generated by that potential.